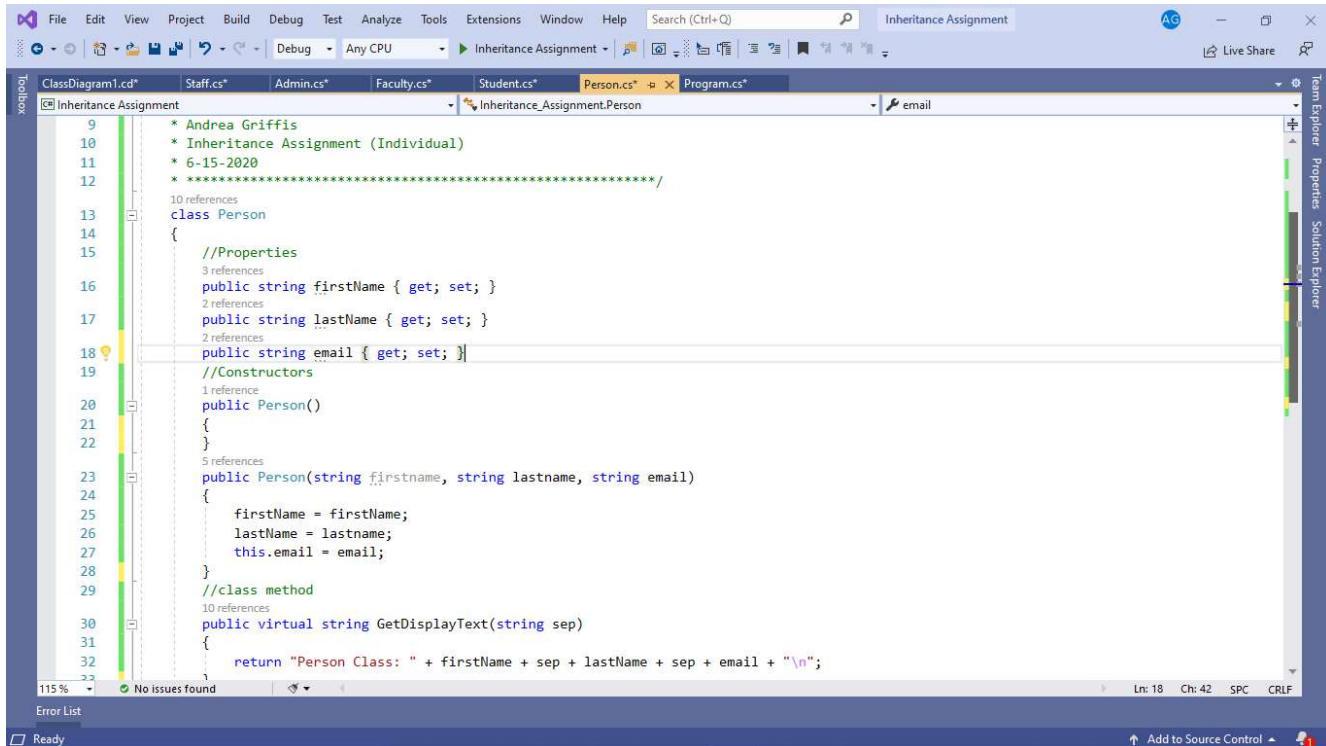


Assignment: Inheritance Assignment (Individual)

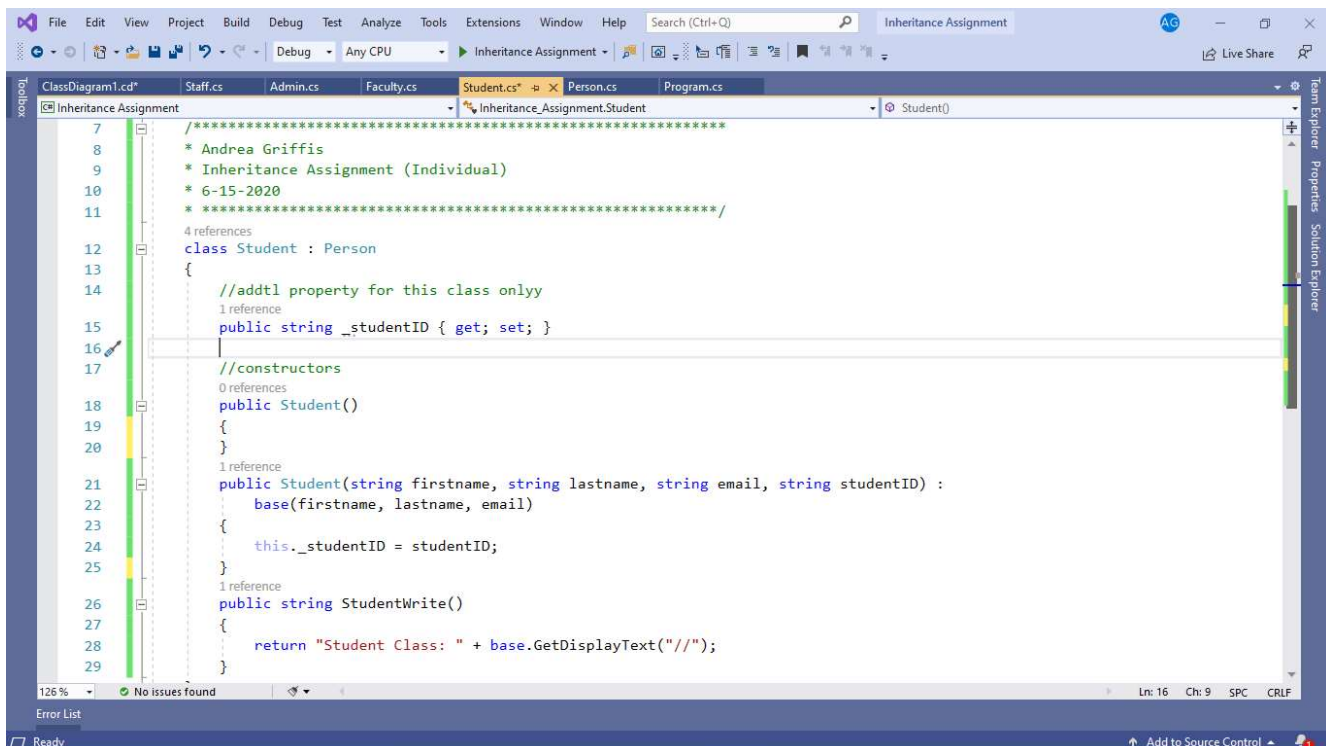
Complete the Person (base), Student (derived), Faculty (derived), Admin (derived), Staff (derived)
Person properties - firstName, lastName, email (type string)

Person method - PersonWrite



```
9  * Andrea Griffis
10 * Inheritance Assignment (Individual)
11 * 6-15-2020
12 * *****/
13 10 references
14 class Person
15 {
16     //Properties
17     3 references
18     public string firstName { get; set; }
19     2 references
20     public string lastName { get; set; }
21     2 references
22     public string email { get; set; }
23     //Constructors
24     1 reference
25     public Person()
26     {
27     }
28     5 references
29     public Person(string firstName, string lastName, string email)
30     {
31         firstName = firstName;
32         lastName = lastName;
33         this.email = email;
34     }
35     //class method
36     10 references
37     public virtual string GetDisplayText(string sep)
38     {
39         return "Person Class: " + firstName + sep + lastName + sep + email + "\n";
40     }
41 }
```

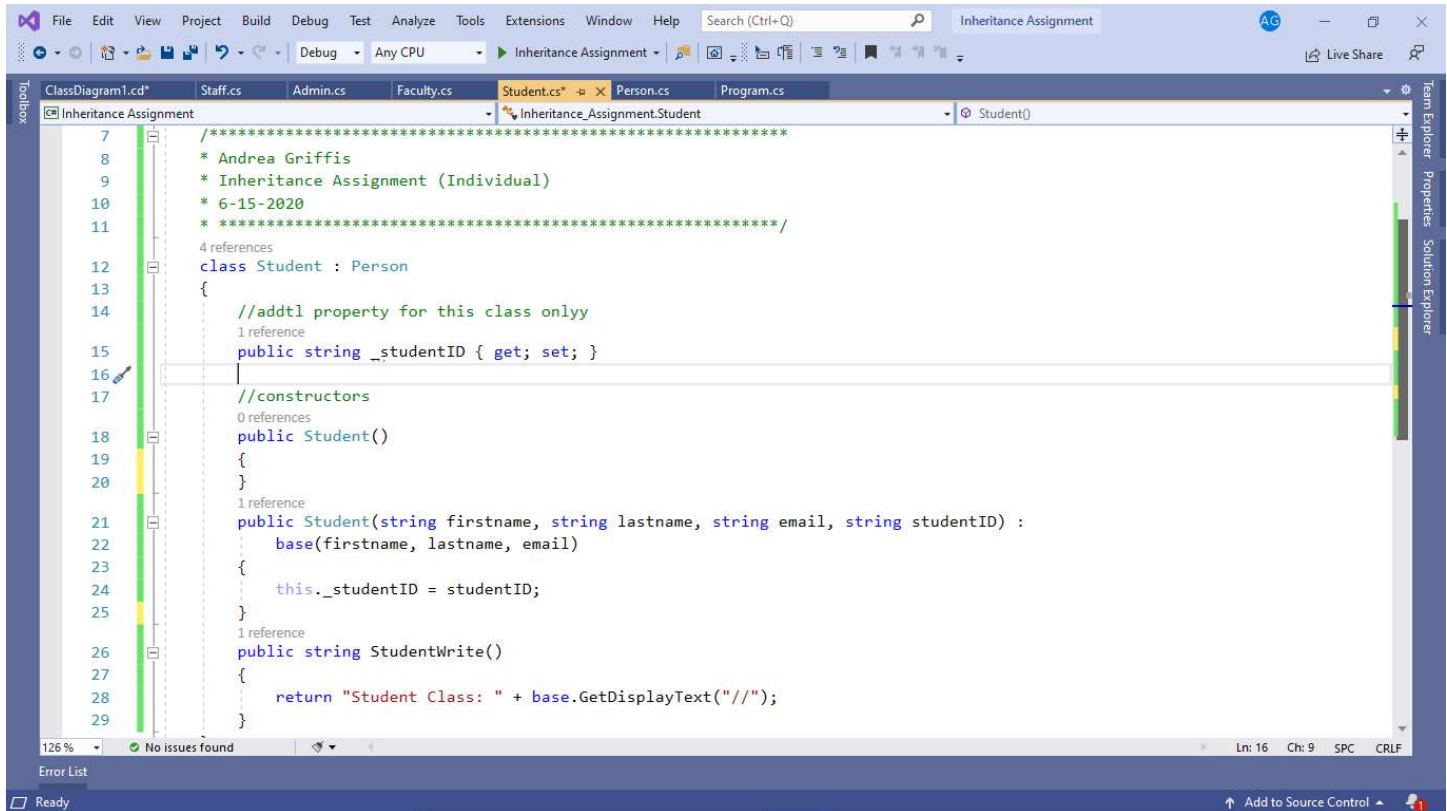
2- Student properties - studentID (private string), use a getter and setter, define 2 constructors
Student method - StudentWrite



```
7  /******/
8  * Andrea Griffis
9  * Inheritance Assignment (Individual)
10 * 6-15-2020
11 * *****/
12 4 references
13 class Student : Person
14 {
15     //addtl property for this class only
16     1 reference
17     public string _studentID { get; set; }
18     //constructors
19     0 references
20     public Student()
21     {
22     }
23     1 reference
24     public Student(string firstName, string lastName, string email, string studentID) :
25         base(firstName, lastName, email)
26     {
27         this._studentID = studentID;
28     }
29     1 reference
30     public string StudentWrite()
31     {
32         return "Student Class: " + base.GetDisplayText("//");
33     }
34 }
```

3- Faculty properties - facultyID (private string), use a getter and setter, define 2 constructors

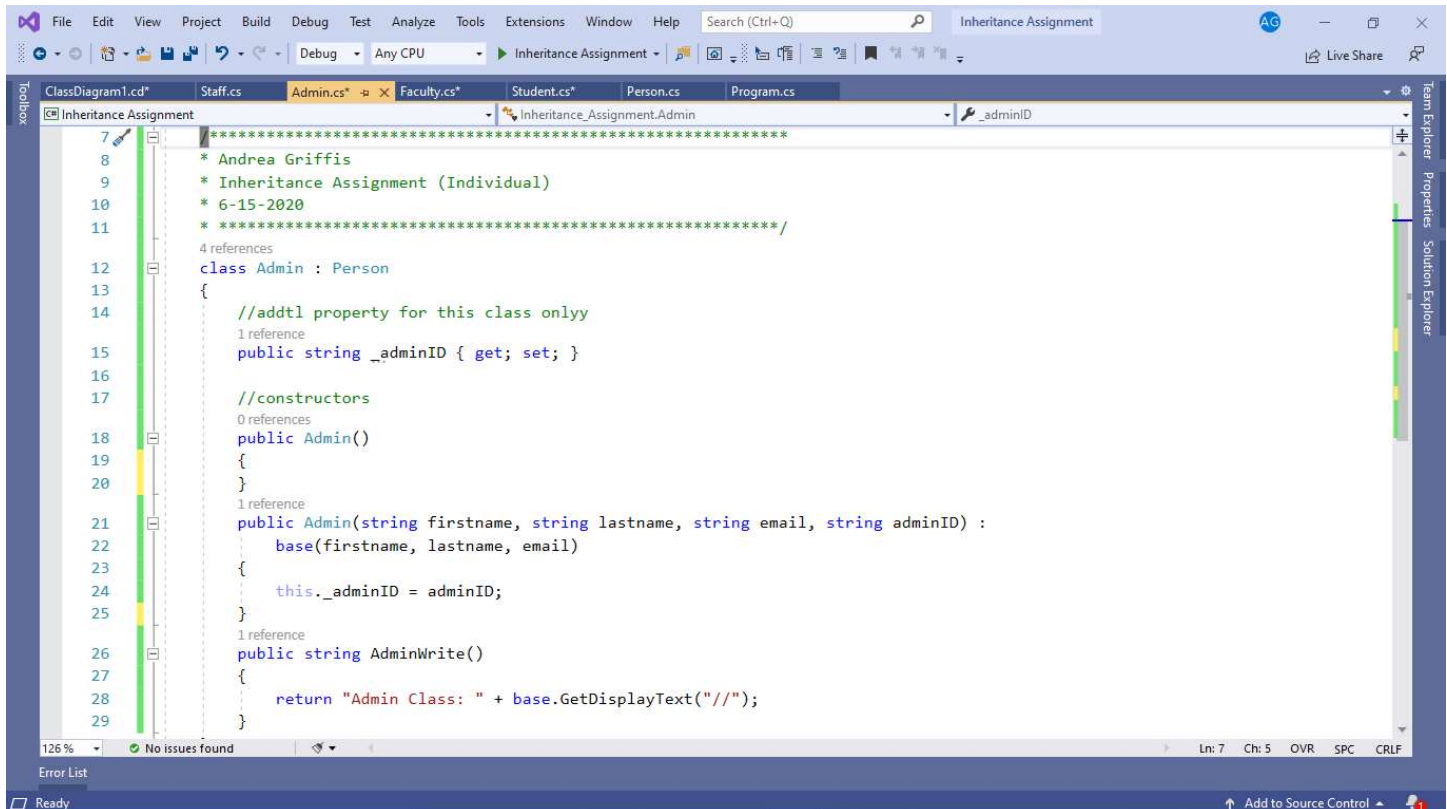
Faculty method - FacultyWrite



```
7  /*****  
8  * Andrea Griffis  
9  * Inheritance Assignment (Individual)  
10 * 6-15-2020  
11 * *****/  
12 4 references  
13 class Student : Person  
14 {  
15     //addtl property for this class only  
16     1 reference  
17     public string _studentID { get; set; }  
18  
19     //constructors  
20     0 references  
21     public Student()  
22     {  
23     }  
24     1 reference  
25     public Student(string firstname, string lastname, string email, string studentID) :  
26         base(firstname, lastname, email)  
27     {  
28         this._studentID = studentID;  
29     }  
30  
31     1 reference  
32     public string StudentWrite()  
33     {  
34         return "Student Class: " + base.GetDisplayText("//");  
35     }  
36 }
```

4- Admin properties - adminID (private string), use a getter and setter, define 2 constructors

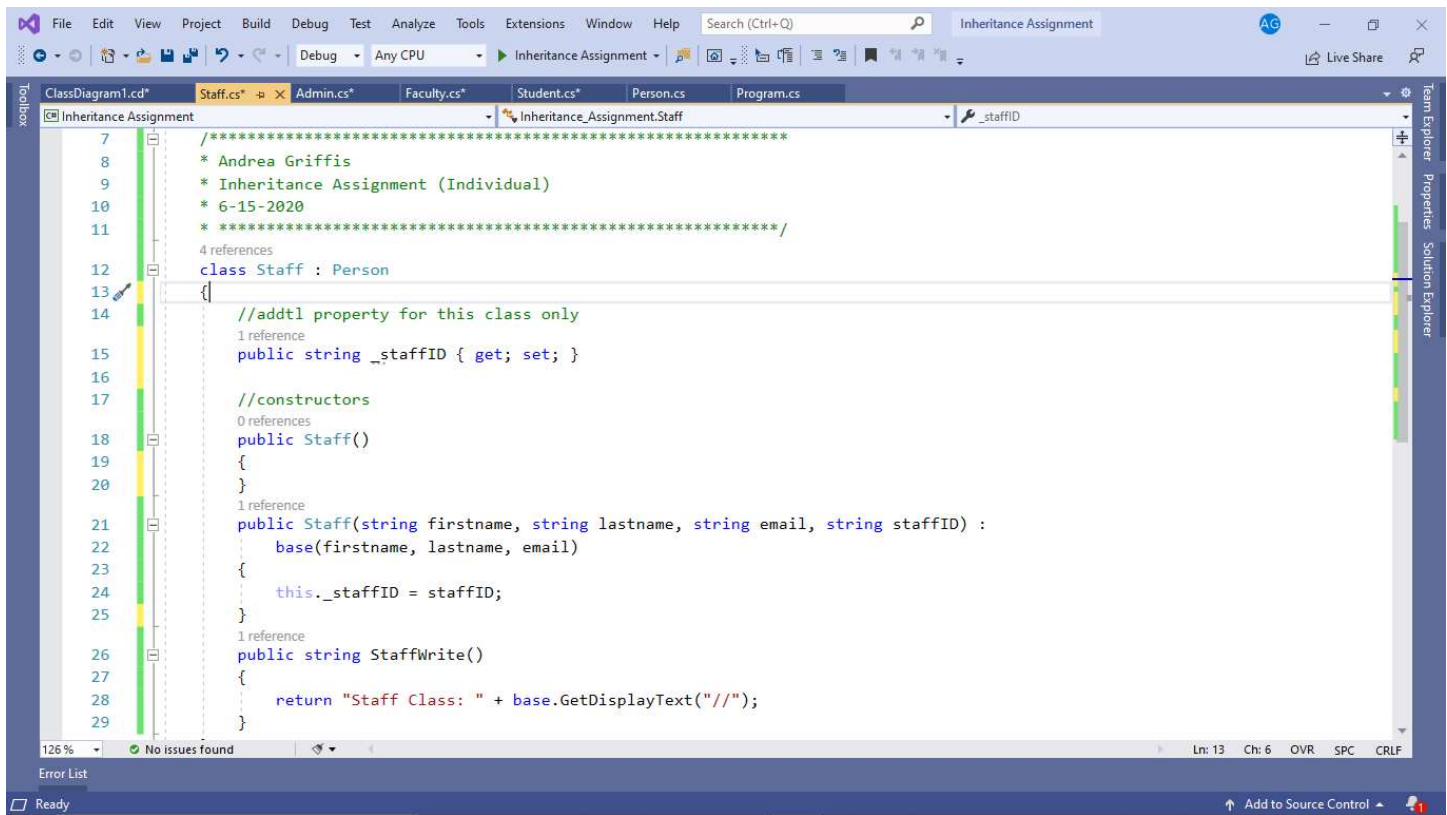
Admin method - AdminWrite



```
7  /*****  
8  * Andrea Griffis  
9  * Inheritance Assignment (Individual)  
10 * 6-15-2020  
11 * *****/  
12 4 references  
13 class Admin : Person  
14 {  
15     //addtl property for this class only  
16     1 reference  
17     public string _adminID { get; set; }  
18  
19     //constructors  
20     0 references  
21     public Admin()  
22     {  
23     }  
24     1 reference  
25     public Admin(string firstname, string lastname, string email, string adminID) :  
26         base(firstname, lastname, email)  
27     {  
28         this._adminID = adminID;  
29     }  
30  
31     1 reference  
32     public string AdminWrite()  
33     {  
34         return "Admin Class: " + base.GetDisplayText("//");  
35     }  
36 }
```

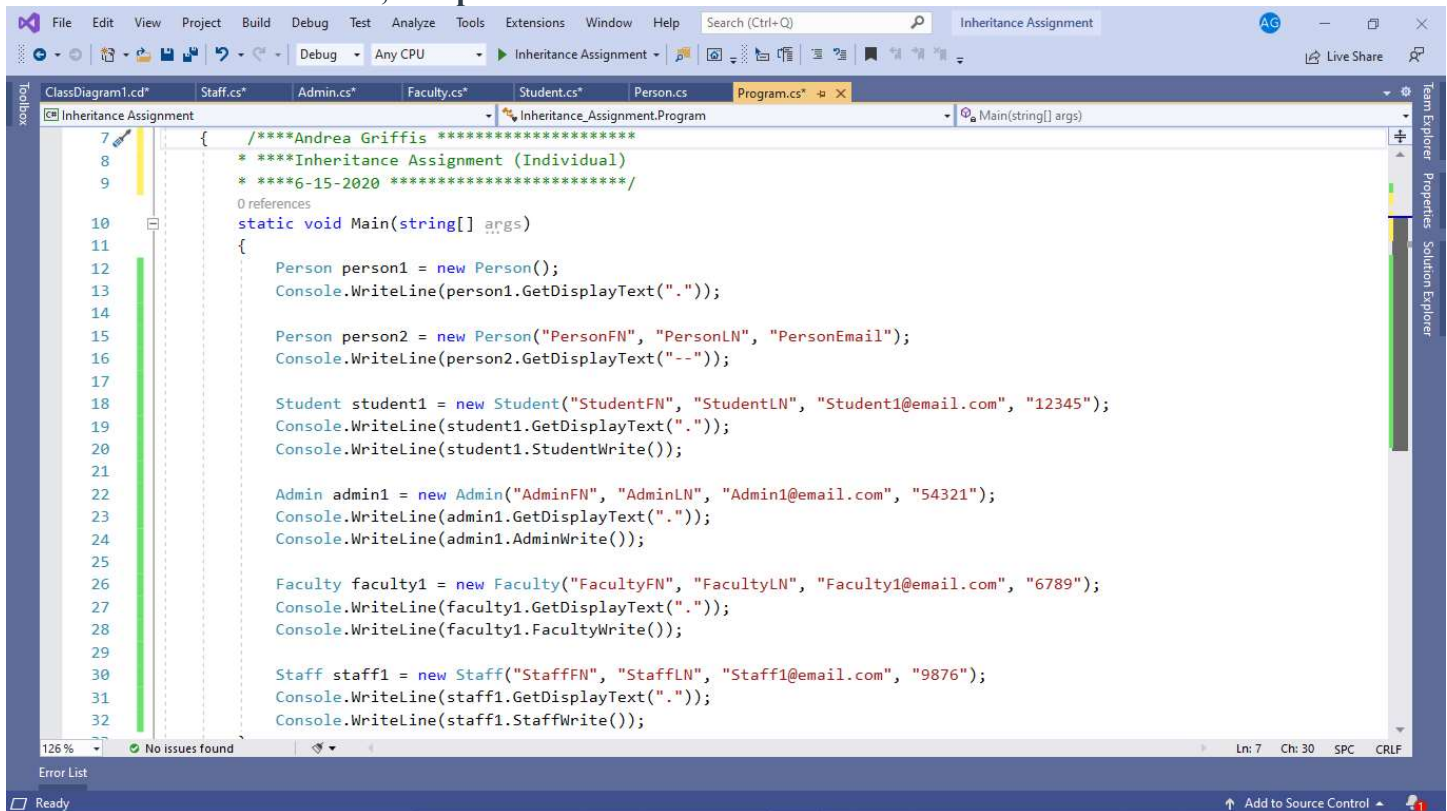
5- Staff properties - staffID (private string), use a getter and setter, define 2 constructors

Staff method - StaffWrite



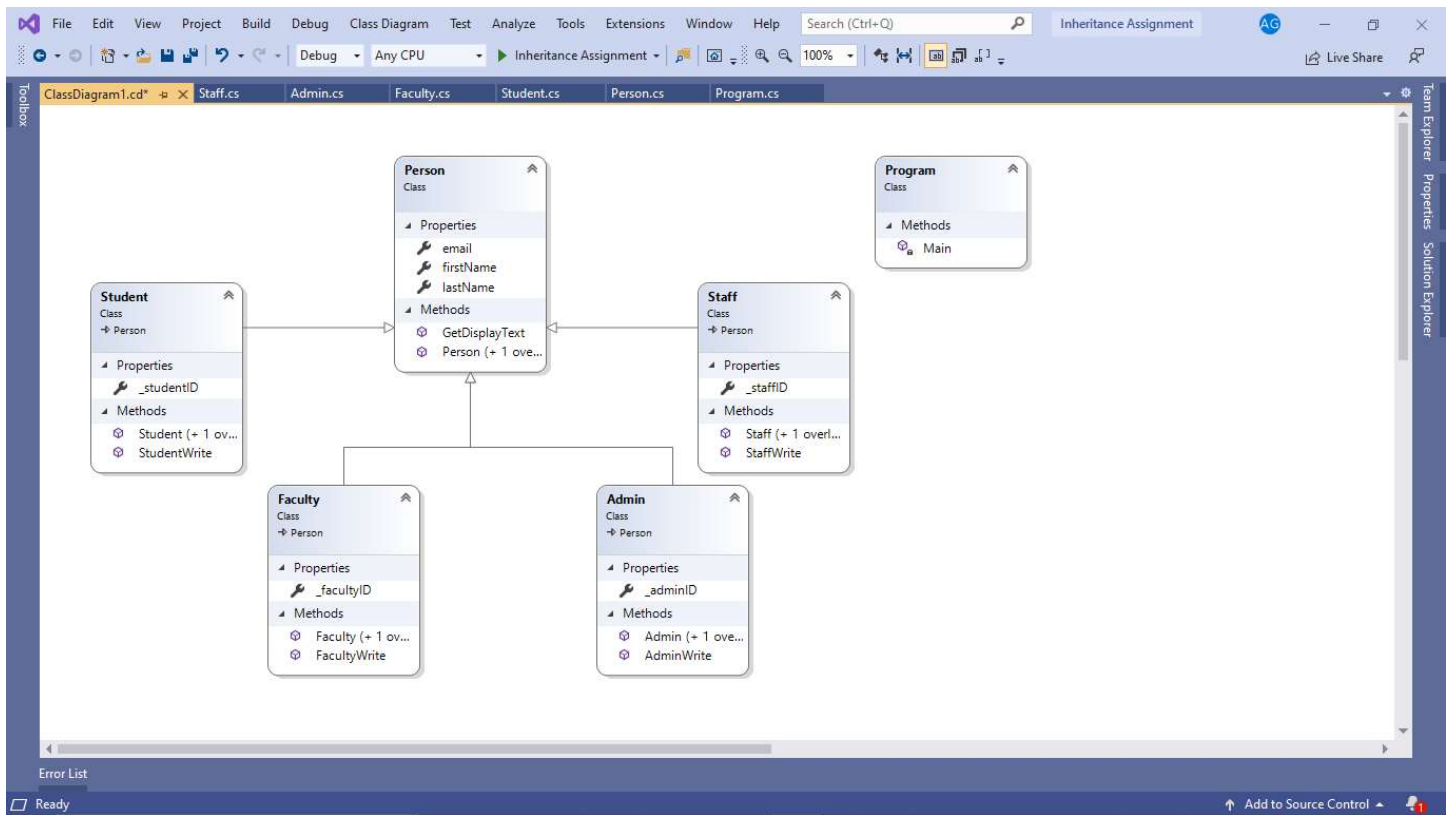
```
7  //*****Andrea Griffis *****
8  * Inheritance Assignment (Individual)
9  * 6-15-2020 *****/
10
11 4 references
12 class Staff : Person
13 {
14     //addtl property for this class only
15     //1 reference
16     public string _staffID { get; set; }
17
18     //constructors
19     //0 references
20     public Staff()
21     {
22     }
23     //1 reference
24     public Staff(string firstname, string lastname, string email, string staffID) :
25         base(firstname, lastname, email)
26     {
27         this._staffID = staffID;
28     }
29     //1 reference
30     public string StaffWrite()
31     {
32         return "Staff Class: " + base.GetDisplayText("/");
33     }
34 }
```

Initialize each of the classes, and print out the contents of each class to show inheritance.



```
7  //****Andrea Griffis ****
8  * Inheritance Assignment (Individual)
9  * 6-15-2020 ****/
10
11 0 references
12 static void Main(string[] args)
13 {
14     Person person1 = new Person();
15     Console.WriteLine(person1.GetDisplayText("."));
16
17     Person person2 = new Person("PersonFN", "PersonLN", "PersonEmail");
18     Console.WriteLine(person2.GetDisplayText("--"));
19
20     Student student1 = new Student("StudentFN", "StudentLN", "Student1@email.com", "12345");
21     Console.WriteLine(student1.GetDisplayText("."));
22     Console.WriteLine(student1.StudentWrite());
23
24     Admin admin1 = new Admin("AdminFN", "AdminLN", "Admin1@email.com", "54321");
25     Console.WriteLine(admin1.GetDisplayText("."));
26     Console.WriteLine(admin1.AdminWrite());
27
28     Faculty faculty1 = new Faculty("FacultyFN", "FacultyLN", "Faculty1@email.com", "6789");
29     Console.WriteLine(faculty1.GetDisplayText("."));
30     Console.WriteLine(faculty1.FacultyWrite());
31
32     Staff staff1 = new Staff("StaffFN", "StaffLN", "Staff1@email.com", "9876");
33     Console.WriteLine(staff1.GetDisplayText("."));
34     Console.WriteLine(staff1.StaffWrite());
35 }
```


Create a Class Diagram for your program using VS Class Diagram/Designer feature. (10 pts)



```
Microsoft Visual Studio Debug Console

Person Class: ..
Person Class: --PersonLN--PersonEmail
Person Class: .StudentLN.Student1@email.com
Student Class: Person Class: //StudentLN//Student1@email.com
Person Class: .AdminLN.Admin1@email.com
Admin Class: Person Class: //AdminLN//Admin1@email.com
Person Class: .FacultyLN.Faculty1@email.com
Faculty Class: Person Class: //FacultyLN//Faculty1@email.com
Person Class: .StaffLN.Staff1@email.com
Staff Class: Person Class: //StaffLN//Staff1@email.com

C:\Users\Student.LABTOPG011860\source\repos\Inheritance Assignment\Inheritance Assignment\bin\Debug\netcoreapp3.1\Inheritance Assignment.exe (process 9764) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .
```